



OPEN

A meta-learning framework to mitigate negative transfer in transfer learning applicable to drug design

Antonia Mera^{1,2}, Martin Vogt^{1,2} & Jürgen Bajorath^{1,2✉}

Data sparseness is a major limiting factor for deep machine learning. In the natural sciences, data distributions are heterogeneous. For instance, in chemistry and early-phase drug discovery, compound and molecular property data are typically sparse compared to data in other fields such as particle physics or genome biology. For machine learning in low-data regimes, approaches such as transfer learning or meta-learning have been introduced. These learning strategies are conceptually related but algorithmically distinct and typically applied independently. They share the common goal of facilitating knowledge transfer between domains with related prediction tasks and varying data availability. We were interested in combining meta- and transfer learning into a coherent framework, primarily for deep learning in cheminformatics. Therefore, we introduce a new meta-learning algorithm designed to complement transfer learning. It identifies an optimal subset of training instances and determines weight initializations for deriving base models that can then be fine-tuned under conditions of data scarcity. Given its ability to identify preferred training samples, the meta-learning algorithm balances negative transfer between source and target domains, which represents a major caveat for transfer learning. In an extensive proof-of-concept application, inhibitors of protein kinases were predicted following data reduction using combined meta- and transfer learning, revealing statistically significant increases in model performance and effective control of negative transfer.

Keywords Deep learning, Low-data regimes, Transfer learning, Meta-learning, Adaptive knowledge transfer, Model generalization, Negative transfer, Cheminformatics

In early-phase drug discovery and design, compound and molecular property data are typically sparse, which often limits meaningful deep machine learning applications. For related prediction tasks, data constraints can be circumvented through the application of methods such as transfer learning, which aims to learn features that are transferable between tasks to compensate for sparse data¹. Transfer learning formally distinguishes between the source domain consisting of one or more tasks that are related to the target domain representing the primary task(s) of interest. For instance, an exemplary transfer learning strategy involves pre-training of a model for a data-restricted task (target domain) on data for a related task (source domain), followed by fine-tuning on the data-restricted tasks¹.

Sequential single-task learning can be further extended using multi-task transfer learning models. While multi-task learning is also applicable without transfer components, it represents a suitable framework for transfer learning. For example, Ye et al.² used a bioactivity data set comprising ligands of 157 target proteins to pre-train a multi-task neural network composed of a feature extractor and a task layer. Weights from pre-training were then transferred to another multi-task neural network derived to predict pharmacokinetic parameters for a set of approved drugs. This transfer learning strategy reached higher prediction accuracy than conventional machine learning models and had better generalization ability². While transfer learning is a method of choice for predictions in low-data regimes, it is generally difficult to confidently select tasks where transfer learning will be superior to other machine learning approaches. For example, although compound activity prediction for members of the same protein family basically represents a suitable task for transfer learning, optimal learning conditions for such predictions might substantially differ³. Therefore, methods are desirable to quantify task similarity and guide the

¹Department of Life Science Informatics and Data Science, LIMES Program Unit Chemical Biology and Medicinal Chemistry, B-IT, Friedrich-Hirzebruch-Allee 5/6, Bonn, Germany. ²Lamarr Institute for Machine Learning and Artificial Intelligence, University of Bonn, Friedrich-Hirzebruch-Allee 5/6, D-53115 Bonn, Germany. ✉email: bajorath@bit.uni-bonn.de

selection of the source domain. For instance, similarity between target and potential source tasks can be assessed based on latent data representations learned by graph neural networks individually pre-trained for each task⁴. This approach is primarily applicable to uniform data such as compounds having a specific activity. In addition, similarity (or distance) scoring is also applicable by combining similarity assessment for data representations such as protein sequence and chemical space embeddings⁵. However, deriving a task similarity measure for multiple tasks or combinations of tasks typically is computationally demanding, emphasizing the need for an adaptive weighting algorithm. Meta-learning is another approach for addressing prediction tasks in low-data regimes⁶. Different from the standard pre-training and fine-tuning scheme of transfer learning encompassing different domains, meta-learning aims to derive models that can effectively adapt to new low-data tasks, without the need for extensive additional training. In a typical meta-learning framework, a base model is trained in an inner loop to address an individual task, while an outer loop optimizes a given “meta-objective” such as learning a weight initialization to improve generalization across tasks⁶. The ability of meta-learning to meet this objective for limited training instances also depends on task similarity⁷.

For neural network architectures, weight distributions for modeling of related tasks can be derived in different ways. For example, the Meta-Weight-Net algorithm was designed to learn sample weights based on their classification loss⁸. Therefore, a shallow neural network uses the loss from a base model prediction of a test instance as input and derives a weight for this instance. Utilizing the loss in a weighting scheme can guide learning based on the hardness of each instance to be classified by the model, as implemented in the AdaBoost algorithms⁹. However, a larger feature space, as can be covered by the new algorithm introduced herein, is likely to contain increasing amounts of important information, especially when addressing multiple tasks, which substantially aids in deriving a weighting scheme. Furthermore, the Model-Agnostic Meta-Learning (MAML) algorithm searches for weight initializations that only require a few gradient steps to train a base model¹⁰. While Meta-Weight-Net and related algorithms derive weights for individual samples, MAML operates on tasks that are expected to support a given target task. For example, MAML has been applied to increase the learning effectiveness and performance of transformer-based chemical language models in predicting potent compounds based on weakly active templates¹¹. However, the MAML approach cannot be applied if source and target tasks lack significant similarity^{12,13}, leading to so-called negative transfer¹⁴, which is known to compromise transfer learning. Negative transfer of task information decreases the performance of a transfer learning model relative to the base model¹⁴. In contrast to MAML, our meta-learning method uses both sample and task information for its unique meta-objective to mitigate negative transfer, which offers opportunities in complementing methods designed to use information from multiple related tasks. Importantly, negative transfer can also take place at the instance level, for example, due to the presence of activity or selectivity cliffs in compound data sets, which the currently available meta-learning frameworks including MAML do not take into account. Moreover, although techniques to mitigate negative transfer have been introduced^{15,16}, there currently is no method available that regulates negative transfer with meta-learning. This provides an opportunity for combining meta- and transfer learning. However, although transfer learning and meta-learning are conceptually related, potential synergies between these approaches for machine learning in low-data regimes have thus far not been explored.

To combine the strengths of transfer and meta-learning, our new meta-learning algorithm specifically complements standard transfer learning. The algorithm combines task and sample information and its unique meta-objective is the optimization of the generalization potential of a pre-trained transfer learning model in the target domain. Therefore, a model is pre-trained in a source domain using weights determined by the meta-learning algorithm. This provides a basis for effective fine-tuning of the transfer learning model in a target domain. The unique feature of the meta-learning algorithm enabling this combined approach is the identification of an optimal subset of source samples for pre-training of the transfer learning model. The ability to optimize training sample selection makes it possible to algorithmically balance negative transfer between the source and target domains. This major limitation of transfer learning is directly addressed for the first time by combining meta- and transfer learning with our new algorithm.

Methods

Compounds, activity data, and molecular representations

For our analysis, a protein kinase inhibitor (PKI) data set was generated. Therefore, protein kinases (PKs) comprising the human kinome¹⁷ and activity data for PKIs were systematically collected from ChEMBL¹⁸ (version 34) and BindingDB¹⁹ and combined according to Vossen et al.²⁰, yielding more than 450,000 PKIs with activity against 461 PKs. The PKI data set was filtered to only contain (assay-independent) K_i values as activity annotations and compounds with a molecular mass of less than 1000 Da. Structures of qualifying compounds were standardized and canonical nonisomeric SMILES strings²¹ were generated using RDKit²². For multiple available K_i values per compound c for a given PK, the geometric mean was calculated if these values met the condition $\frac{K_i^{c\max}}{K_i^{c\min}} \leq 10$ (if not, the measurements were discarded). The final curated data set contained 7098 unique PKIs with activity against 162 PKs and a total of 55,141 PK annotations.

For activity-based classification, K_i values were transformed into a binary (active/inactive) format by applying a potency threshold of 1000 nM. Accordingly, compounds with a K_i value below or above this threshold were labeled as active and inactive, respectively. Of note, the choice of this threshold was motivated by practical medicinal chemistry where PKIs with 1000 nM activity are typically considered inactive, given the need for at least low-nanomolar PKIs for drug development.

For transfer and meta-learning, 19 PKs with at least qualifying 400 PKIs and 25–50% of these PKIs classified as active were selected. Table 1 summarizes these data sets. The total number of PKIs and the number of classified actives per set ranged from 474 to 1028 and 151–363, respectively.

Abbreviation	Protein kinase	# PKIs	# Actives
	AGC Ser/Thr protein kinase family		
PKN2	Serine/threonine-protein kinase N2	699	183
PRKX	cAMP-dependent protein kinase catalytic subunit PRKX	733	197
	Tyr protein kinase family		
BLK	Tyrosine-protein kinase Blk	635	216
	CAMK Ser/Thr protein kinase family		
DAPK3	Death-associated protein kinase 3	892	239
MELK	Maternal embryonic leucine zipper kinase	564	171
MKNK2	MAP kinase-interacting serine/threonine-protein kinase 2	665	195
STK17A	Serine/threonine-protein kinase 17 A	576	186
	CMGC Ser/Thr protein kinase family		
CDK8	Cyclin-dependent kinase 8	474	195
CLK2	Dual specificity protein kinase CLK2	855	231
DYRK1A	Dual specificity tyrosine-phosphorylation-regulated kinase 1 A	1028	336
HIPK2	Homeodomain-interacting protein kinase 2	669	204
HIPK4	Homeodomain-interacting protein kinase 4	599	151
	STE Ser/Thr protein kinase family		
MAP4K2	Mitogen-activated protein kinase kinase kinase kinase 2	858	245
MAP4K4	Mitogen-activated protein kinase kinase kinase 4	971	363
MAP4K5	Mitogen-activated protein kinase kinase kinase kinase 5	666	238
MINK1	Misspliced kinase 1	644	165
SLK	STE20-like serine/threonine-protein kinase	677	221
STK3	Serine/threonine-protein kinase 3	796	204
	TKL Ser/Thr protein kinase family		
LIMK1	LIM domain kinase 1	850	227

Table 1. Protein kinase inhibitor sets.

As a molecular representation for machine learning, the extended connectivity fingerprint²³ with bond diameter of 4 (ECFP4) and a constant size of 4096 bits was generated from SMILES strings of compounds using RDKit.

Method formulation

Standard transfer learning leverages knowledge from a source domain with abundant data to improve learning in a low-data target domain. As discussed above, performance in the target domain can be compromised if the two data domains are not sufficiently similar, causing negative transfer¹⁴.

The meta-learning algorithm introduced herein aims to mitigate negative transfer. Therefore, a meta-model must derive weights for the source data points adjusting the relative contributions of samples during pre-training of a base model (Fig. 1). We apply the method to PKI data sets since compound information for multiple PKs can be readily transferred to a PK with reduced compound data, as detailed below.

First, two data sets are specified including a target data set (inhibitors of a data-reduced PK): $T^{(t)} = \{(x_i^t, y_i^t, s^t)\}$

and a source data set (containing PKIs of multiple PKs excluding the target PK):

$$S^{(-t)} = \{(x_j^k, y_j^k, s^k)\}_{k \neq t}$$

Here, x represents the molecule, y is the label, and s is a protein sequence representation.

Next, we define the two models for the meta-learning framework as follows:

The base model f with parameters θ for classifying active vs. inactive compounds is trained on the source data $S^{(-t)}$ with a weighted loss function, in which the weights correspond to the weight predictions of a meta-model g for each data point. For the target data set T , the base model predicts the binary activity states of the compounds in the target training data set. From the predicted activity states, the validation loss is calculated, adding a second layer of optimization using the validation loss to update the meta-model.

The meta-model g with parameter φ predicts weights for training data of the base model. For a source data set $S^{(-t)}$ composed of K PKs together with their ligands $\{(x_j^k, y_j^k, s^k)\}_{k \neq t}$, the meta-model assesses how informative the j^{th} compound of the k^{th} PK is for the target PK t by predicting a weight w_j^k . Weight predictions are limited to the interval $(0,1)$ by the sigmoid function to avoid extreme predictions during training.

The weights predicted by the meta-model are then used to train the base model with the following weighted binary cross entropy

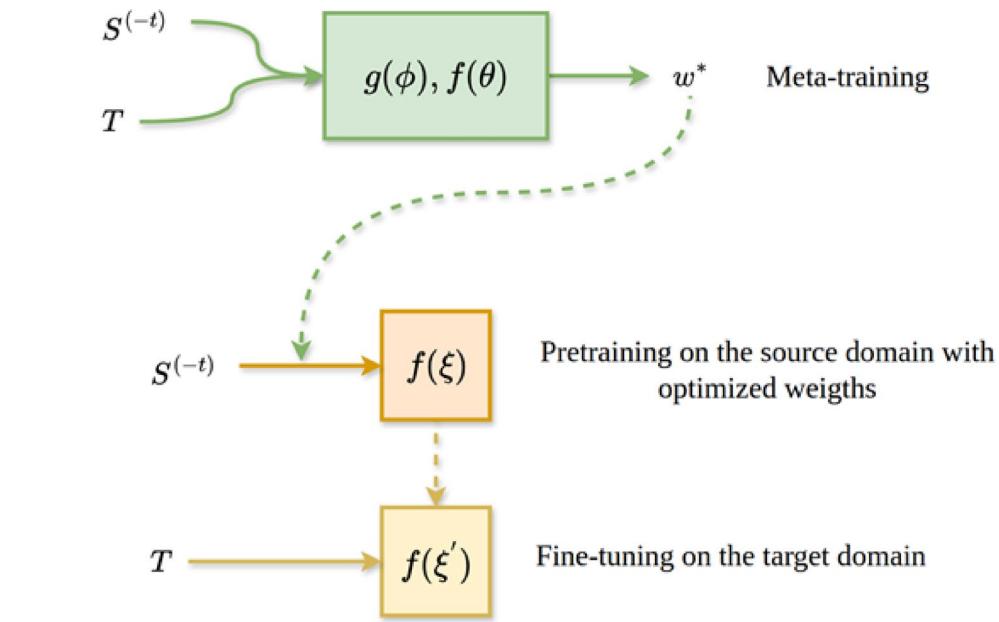


Fig. 1. Meta-learning principles. The meta-training procedure uses a base model f and meta-model g to optimize source $S^{(-t)}$ sample weights w . Parameters of the base model f are re-initialized and adjusted using the weighting scheme learned by the meta-model. Fine-tuning is then applied to the target domain T .

$$L_{train} = \frac{\sum_{k=1, k \neq t}^K \sum_{j=1}^{N_k} g(x_j^k, s^k; \varphi) \text{BCE}(y_j^k, f(x_j^k; \theta))}{\sum_{k=1, k \neq t}^K \sum_{j=1}^{N_k} g(x_j^k, s^k; \varphi)} \quad (1)$$

where $\text{BCE}(u, \hat{u}) = u \log(\hat{u}) + (1-u) \log(1-\hat{u})$.

The central idea of the method is optimizing w_j^k values and training the base model on the source data $S^{(-t)}$ according to Eq. 1. Accordingly, the base model should improve performance on the target data T compared to its counterpart model that is trained with the unweighted loss. For weight optimization, the validation loss of the base model is estimated based on the target training data T and used to train the meta-model. The loss function for the validation loss is the standard binary cross-entropy (BCE) loss and computed as:

$$L_{val} = \frac{\sum_{i=1}^{N_t} \text{BCE}(y_i^t, f(x_i^t; \theta))}{N} \quad (2)$$

Training of the meta-model is based on these two losses. First, the validation loss with respect to parameter φ needs to be computed. Since L_{val} is only a function of θ , the indirect dependence of φ on θ is added and the chain rule is applied. The meta-gradients are of the general form:

$$\frac{\partial L_{val}}{\partial \varphi} = \frac{\partial L_{val}}{\partial \theta} \cdot \frac{\partial \theta}{\partial \varphi} \quad (3)$$

where $\partial \theta$ can be derived by differentiating the backpropagation of the base model with respect to the meta-model's φ parameters.

$$\partial \theta = -\eta_\theta \cdot \frac{\partial L_{train}}{\partial \theta} \Rightarrow \frac{\partial \theta}{\partial \varphi} = -\eta_\theta \cdot \frac{\partial^2 L_{train}}{\partial \theta \partial \varphi} \quad (4)$$

Combining Eqs. 3 and 4 yields:

$$\frac{\partial L_{val}}{\partial \varphi} = \frac{\partial L_{val}}{\partial \theta} \cdot \left(-\eta_\theta \cdot \frac{\partial^2 L_{train}}{\partial \theta \partial \varphi} \right) \quad (5)$$

The backpropagation on the meta-model is then formulated as:

$$\partial \varphi = -\eta_\varphi \left(-\eta_\theta \frac{\partial L_{val}}{\partial \theta} \frac{\partial^2 L_{train}}{\partial \theta \partial \varphi} \right) = +\eta'_\varphi \frac{\partial L_{val}}{\partial \theta} \frac{\partial^2 L_{train}}{\partial \theta \partial \varphi} \quad (6)$$

with learning rates η_φ and η_θ that are combined to $\eta' = \eta_\varphi \cdot \eta_\theta$. For practical purposes, η' is treated as an independent parameter. Algorithm 1 summarizes the meta-training procedure and Fig. 2 illustrates the meta-learning framework.

Required: Source data \mathbf{B}^t , target data \mathbf{B}^v

- 1: Initialize meta-model, base model, number of epochs N
- 2: **for** $i = 1$ **to** N **do**
- 3: $B^t, B^v \leftarrow \text{shuffle}(B^t, B^v)$
- 4: **for** b^t **in** \mathbf{B}^t **do**
- 5: $w \leftarrow \text{meta_model}(b^t, s^k)$
- 6: $\hat{y}_t \leftarrow \text{base_model}(b^t)$
- 7: $L_{\text{train}} \leftarrow \frac{\sum w \cdot \text{BCELoss}(y_t, \hat{y}_t, \text{reduction}=\text{none})}{\sum w}$
- 8: Compute gradient $\frac{\partial L_{\text{train}}}{\partial \theta}$
- 9: $\hat{y}_v \leftarrow \text{base_model}(B^v)$
- 10: $L_{\text{val}} \leftarrow \frac{\sum \text{BCE}(y_v, \hat{y}_v, \text{reduction}=\text{none})}{n}$
- 11: Compute gradient $\frac{\partial L_{\text{val}}}{\partial \theta}$
- 12: Compute meta-gradient $\frac{\partial L_{\text{val}}}{\partial \theta} \frac{\partial^2 L_{\text{train}}}{\partial \theta \partial \phi}$
- 13: Update meta-model
- 14: $w' \leftarrow \text{meta_model}(b^t, s^k)$
- 15: $L'_{\text{train}} \leftarrow \frac{\sum w' \cdot \text{BCE}(y_t, \hat{y}_t, \text{reduction}=\text{none})}{\sum w'}$
- 16: Compute and accumulate gradients $\frac{\partial L'_{\text{train}}}{\partial \theta}$
- 17: Update base model

Algorithm 1. Meta-training.

Noisy meta-gradients

Numerical second-order gradients have high variance, which can destabilize meta-training and hinder its convergence. For preventing limited convergence, first-order approximations of the meta-gradients are often generated, which tends to reduce the accuracy of model updates¹⁰. We address unstable meta-training by adjusting the size of mini-batches, representing a factor known to influence the variance of gradients²⁴. For small target data sets, training can take place in a single batch. For large source data sets, memory capacity is a serious constraint. Therefore, a different approach is applied by accumulating gradients through batches and updating the model only at the end of an epoch (Algorithm 1), thus effectively eliminating batch size as a significant hyperparameter. This algorithmic modification reduces the overall noise of the first-order gradients used to update the base model. Given the dependency of the meta-gradients on the backpropagation of the base model (Eq. 5), meta-training fluctuations are then stabilized.

Monte Carlo dropout for preventing overfitting

In addition to noisy meta-gradients, overfitting is a common problem in heavily parameterized deep neural networks trained on scarce data. A widely used method for controlling overfitting is so-called dropout, that is, neurons are randomly deactivated during training with a pre-defined probability²⁵. Monte Carlo dropout refers to neuron deactivation enabled during testing and is used to produce stochastic predictions for uncertainty estimates. It was shown to approximate Bayesian inference at low computational cost²⁶ and further increase the predictive performance of a deep neural network compared to standard dropout²⁷.

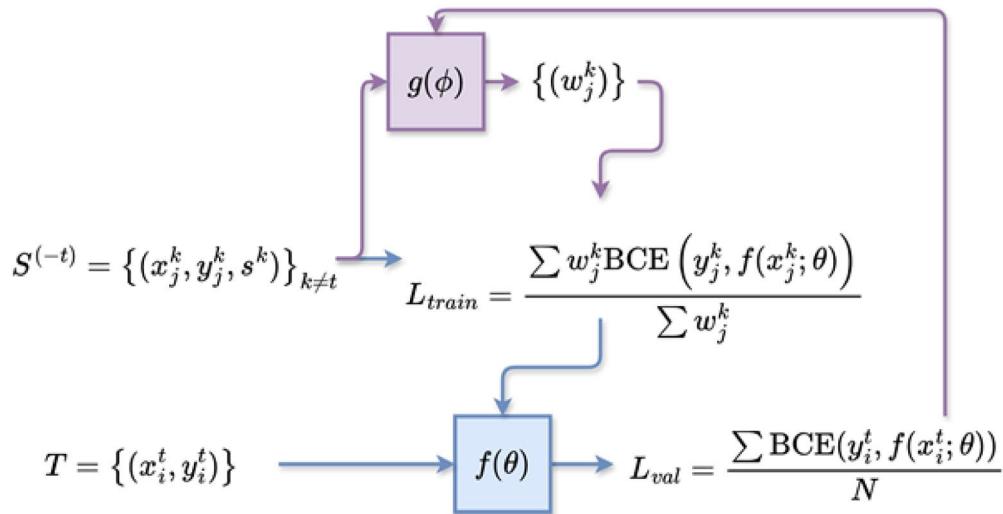


Fig. 2. Meta-learning framework. The base model f is trained using the source data S with the weighted loss L_{train} and validated using the training target data set T with the loss function L_{val} . The validation loss is used for updating the meta-model g and optimizing the predicted weights.

Monte Carlo dropout is incorporated into the meta-model neural network to mitigate overfitting. Therefore, during the pre-training phase of a base model, weights are sampled from the meta-model and used as a part of the weighted loss function (Eq. 1). The number of samples drawn for each compound in the source data set is equal to the number of pre-training epochs. The process of sampling creates a distribution of weights throughout the training phase. The standard deviation of this distribution is directly influenced by the dropout value²⁸. High values can lead to weighted predictions that vary significantly between different epochs, hence complicating the learning process. The dropout rate was selected through grid search optimization and set to 0.2. Higher dropout rates led to a reduction in model performance (Supplementary Fig. 1). The process of training with the weighted loss is summarized in Algorithm 2.

```

1: for  $b^t$  in  $B^t$  do
2:    $x_t, y_t \leftarrow \text{shuffle}(b^t)$ 
3:   meta_model.enable_dropout()
4:    $w \leftarrow \text{meta\_model.sample\_weights}(b^t)$ 
5:    $\hat{y}_t \leftarrow \text{base\_model}(b^t)$ 
6:    $L_{train} \leftarrow \frac{\sum w \cdot \text{BCE}(y_t, \hat{y}_t, \text{reduction}=\text{none})}{\sum w}$ 
7:   Compute gradient  $\frac{\partial L_{train}}{\partial \theta}$ 
8:   Update base model

```

Algorithm 2. Pre-training with Monte Carlo dropout.

Analysis set-up

To evaluate the efficiency of our meta-learning algorithm in preventing negative transfer, we compare the method with standard transfer learning. In both cases, a base model with the same architecture is pre-trained on the source data set $S^{(-t)}$ and fine-tuned on the target data set T using the same fine-tuning protocol. However, the loss functions applied during pre-training with the source data differ. For meta-learning, the base model is trained using weighted BCE loss (with weights predicted by the meta-model) whereas standard transfer learning utilizes unweighted BCE loss for training.

The performance of the models is evaluated using two calculation settings. In the first setting, the compound spaces of the source and target domains overlap, that is, they contain compounds that have K_i measurements for

target PK T as well as for PKs in the source data set $S^{(-t)}$. In the second setting, compounds shared by the two domains are removed from the source data set $S^{(-t)}$ resulting in non-overlapping compound sets:

$$S^{(-t, -\cap)} = \{(x^k, y^k, s^k) \in S \mid k \neq t \text{ and } x^k \notin X^T\} \quad (7)$$

Removal of the shared compounds increases the chemical distance between the source and target domain, resulting in increasing challenges for standard transfer learning, which are addressed by varying hyperparameter settings, as described in the next section.

To study differences in the information transfer from the source to the target domain, random forest (RF) models are derived as an additional control using scikit-learn²⁹ with default parameter settings. In contrast to standard transfer and meta-learning, RF models are exclusively trained on the target domain. Notably, feedforward neural networks were also evaluated as a control, but had lower performance compared to RF. The performance of standard transfer and meta-learning relative to the neural network control model is reported in Supplementary Fig. 2.

Model architectures and hyperparameters

The base model is a feedforward neural network taking ECFP4 representations of compounds as input and producing binary class label predictions (active vs. inactive) as output (Fig. 3). The architecture of the meta-model is slightly more complex, modular, and depends on two types of input data including (i) ECFP4 of a compound (PKI) and (ii) a one-hot encoded representation of the sequence of the associated protein (PK). The modular meta-model network processes these two inputs independently in parallel before the latent representations are concatenated and further processed in additional layers. The sigmoid activation function is applied to the output of the last layer resulting in a predicted weight for the PKI-PK input pair (Fig. 3).

The first two layers of the base model form a so-called bottleneck where the input is projected to a lower-dimensional space and then back to the original dimensionality. This architectural feature enables the construction of a representation containing features characteristic of PKs for learning of PK-dependent data patterns. To retain the learned patterns after pre-training, the bottleneck feedforward layers and subsequent normalization layers³⁰ are frozen, while the following layers are subjected to fine-tuning. This strategy was found to be beneficial for transfer learning using small data sets³¹.

For base model and meta-model training, a learning rate of 10^{-5} is applied in combination with the Adam optimizer³². The batch size of the source data set ranges from 512 to 4096, depending on the data set size (and GPU memory constraints). The number of meta- and pre-training epochs is set to 200 and 150, respectively, for the first calculation setting (see above) and to 100 and 100, respectively, for the second setting. In both cases, fine-tuning is carried out over 100 epochs. The base model and meta-model are implemented with PyTorch³³. Notably, data samples with initially assigned weights close to zero are best excluded from pre-training since they

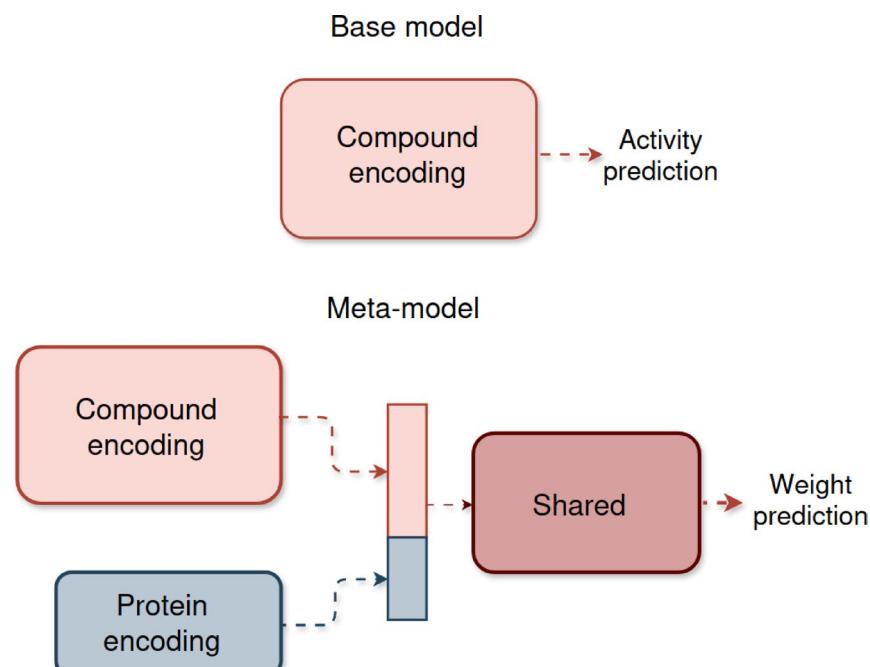


Fig. 3. Model architectures. Shown is a schematic representation of the base model and meta-model. The base model is a feedforward neural network that takes a compound fingerprint as input and predicts the class label of this compound. The meta-model is a modular neural network that takes a compound fingerprint and a one-hot encoded representation of the target protein as input and predicts a weight for this compound-protein pair.

do not significantly contribute to the loss. Frequent occurrence of such data points decreases the effective batch size, which often introduces noise. As training proceeds, weight distributions tend to narrowly center around values of 0 and 1, with decreasing number of values close to 0.5 (Supplementary Fig. 3). A suitable threshold for excluding insignificant samples is chosen depending on the weight distribution observed after training. Here, a weight threshold of 0.05 is applied in the first calculation setting and a threshold of 0.20 in the second setting (given the reduced number of pre- and meta-training epochs).

Pre-training and fine-tuning

For each PK target t , two models are pre-trained on the source domains $S^{(-t)}$ containing a total number of 54,113–54,667 PK-PKI interactions (depending on the target). These models include the standard transfer learning model with uniform weights and the transfer learning model with weights obtained from meta-learning. The models are then fine-tuned on the data-reduced PK target using only 50 training PKIs obtained by stratified sampling from $T^{(t)}$ such that the overall class imbalance (active vs. inactive) is accounted for during training. Of note, test calculations were also carried out for 25 and 100 training PKIs. Sets of 100 training PKIs were already sufficiently large such that standard transfer learning had only marginal effects (Supplementary Fig. 4). On the other hand, using only 25 training PKIs resulted in a significant decrease in performance compared to the use of 50 PKIs, while similar trends were observed (Supplementary Fig. 5). The fine-tuned models are evaluated using the remaining PKIs for PK t .

Given the confined number of fine-tuning training compounds, statistically relevant differences in model performance might be observed in individual trials, depending on the composition of the fine-tuning sets. Therefore, a total of 50 independent fine-tuning trials were carried out for each target to ensure statistical robustness of the results. In each trial, fine-tuning test compounds were ranked according to the probability of activity produced by a model.

Model evaluation

The compound rankings are evaluated using the receiver operating characteristic (ROC) curve³⁴. The ROC curve compares the true positive and false positive rates across the ranking. The area under the ROC curve (AUC), ranging from 0 to 1, is calculated to quantify model performance. An AUC value of 0.5 indicates random classification and ranking whereas a value of 1 indicates a perfect classification (that is, all true positives are ranked higher than the first false positive). AUC value distributions are represented and compared using box plots. In addition, differences in AUC values between the meta- and control models are compared on a trial-by-trial basis using the non-parametric Wilcoxon signed-rank test to assess statistical significance.

To evaluate the effectiveness of the meta-learning approach in controlling negative transfer (see above), the mathematical definition of the negative transfer gap (NTG) by Wang et al.¹⁵ is applied. For a model f trained only on the target domain and another model h trained on the source domain and fine-tuned on the target domain, we define the negative transfer index (NTI) as follows:

$$NTI = \frac{1}{n} \sum_{i=1}^N \text{auc}(y, f(x; T)) - \frac{1}{n} \sum_{i=1}^N \text{auc}(y, h(x; T, S)) \quad (8)$$

where N is the number of trials and auc a function to estimate the AUC value. Accordingly, positive NTI values indicate the presence of negative transfer.

Results

Methodological concept

The major aim of our meta-learning approach is supporting transfer learning in low-data regimes. The basic idea is combining compound and target protein information to further refine weighting schemes for deep learning. By design, this approach is generally applicable to molecular property predictions such as activity-based compound classification. Therefore, models can be pre-trained in a source domain using weights for compound-target pairs determined by meta-learning. This provides an extended basis for fine-tuning using ligands of a new target protein. So-derived models are compared to standard models pre-trained and fine-tuned in the absence of meta-learning.

Specifically, the meta-learning method introduced herein identifies an optimal subset of source samples for pre-training of a base model. Fine-tuning is then performed on target data sets as in standard transfer learning. The meta-objective optimized during meta-training is the generalization performance of the pre-trained model on the target domain. Importantly, by operating at the instance level, negative transfer can be adaptively minimized for each target while increasing the generalization ability of the model. Compared to other meta-learning frameworks, the new algorithm has a number of unique features, is integrated with transfer learning, and strongly emphasizes balancing of negative transfer. Therefore, direct comparisons with other available meta-learning approaches are not possible. For example, the major limitation of the MAML framework is its vulnerability to negative transfer. Therefore, we focus on providing proof-of-principle for the combined meta- and transfer learning approach and quantitative comparisons with standard transfer learning.

For proof-of-concept, we have chosen PKs and their inhibitors as a test system, for several reasons: PKs are closely related and for many PKs, large numbers of PKIs are available. Most currently available PKIs are directed against the ATP cofactor binding site in PKs that is largely conserved across the human kinome^{35,36}. Accordingly, these ATP site-directed PKIs tend to be similar, represent a structural continuum, and are frequently active against more than one PK (multi-PK inhibitors)³⁶. A subset of ~20–30% of currently available PKIs is known to be active against multiple PKs, giving rise to shared compounds in the source and target domains. Furthermore, for PKIs, practically relevant thresholds of activity can be applied to differentiate between related active (potent)

and inactive (weakly potent) compounds. This presents challenges for PKI activity predictions compared to compound data sets where the negative (inactive) class is composed of randomly assembled (and thus irrelevant) compounds. Moreover, given the wealth of available PKs and PKIs, sufficient numbers of PKs can be converted into low-data targets for modeling by selecting small subsets of actives. Notably, PKI data have been used before to evaluate meta-learning with a framework closely related to MAML to predict inhibitors for low-data PKs³⁷. For predicting PKIs, meta-learning was also combined with a graph-attention neural network learning a shared weight initialization for models covering a diverse set of PKs³⁸.

Models for data sets with shared source and target compounds

We first investigated the effects of meta-learning when the source and target domains shared compounds. This was due to the presence of multi-PK inhibitors and generally favored standard transfer learning. For each of the 19 target PKs in Table 1, models were pre-trained in the presence and absence of weights from meta-learning.

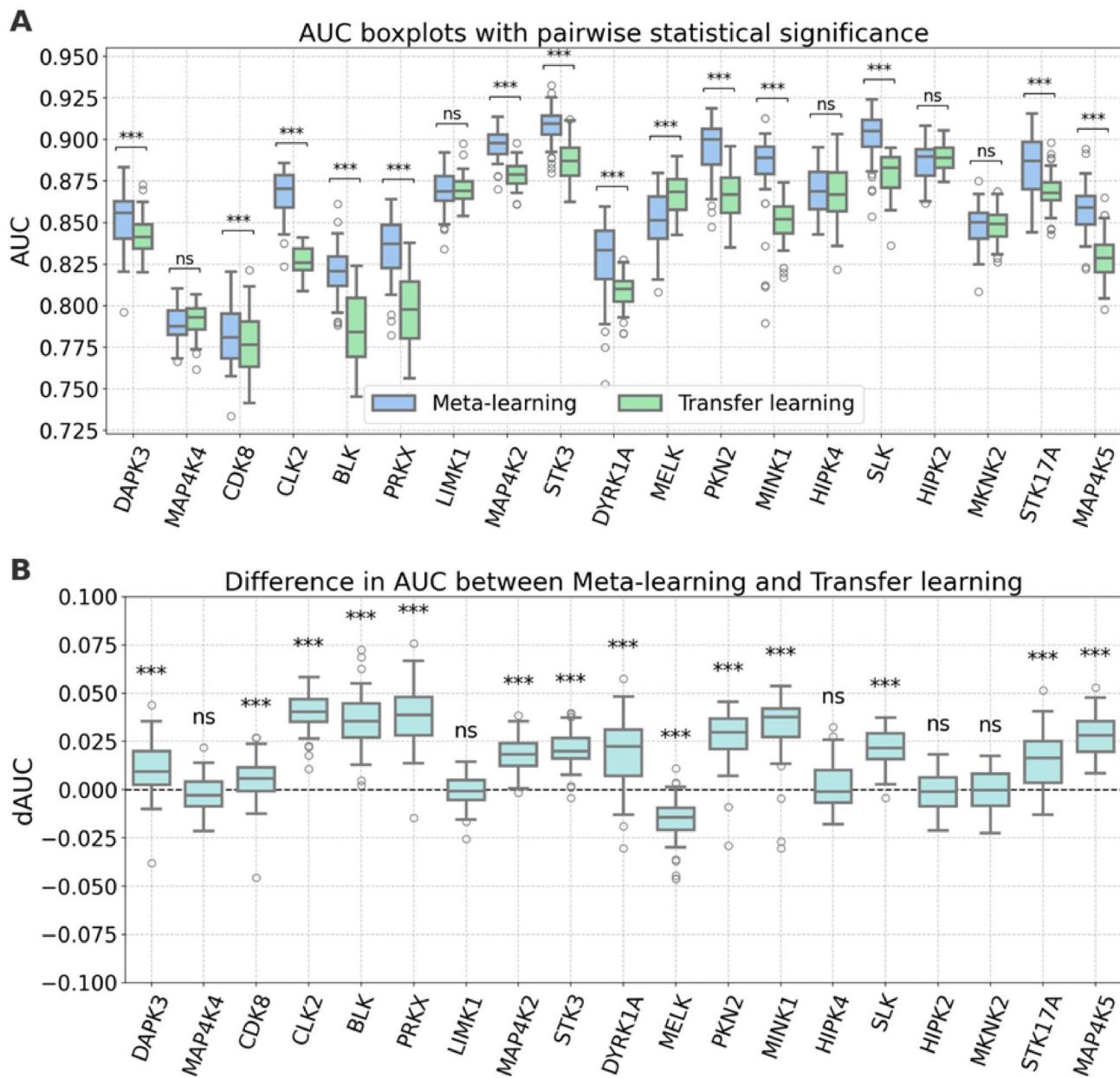


Fig. 4. Model performance for the first calculation setting. (A) Shown are AUC boxplots (box: 1st quartile, median, 3rd quartile; whiskers: ± 1.0 x interquartile range; dots: outliers) for 50 independent trials of meta- and standard transfer learning models for the 19 target PKs. Statistical significance is indicated by asterisks: $0.05 < p\text{-value} \leq 1$: ns (no statistical significance), $0.01 < p\text{-value} \leq 0.05$: *, $0.001 < p\text{-value} \leq 0.01$: **, and $p\text{-value} \leq 0.001$: ***. (B) Boxplots representing the distribution of differences in AUC (dAUC) on a per-trial basis are shown. Positive values indicate improved performance of meta-learning compared to standard transfer learning models.

fine-tuned, and compared. In activity-based classification, median AUC values of the models ranged from ~ 0.75 to ~ 0.90 , depending on the PK target, and independent trials often produced broad AUC value distributions (Fig. 4A). Hence, model performance was overall promising, but heterogeneous, leaving room for improvements. This was anticipated because designated active and inactive compounds represented a continuum of PKIs instead of discrete compound classes. However, in the presence of shared source and target compounds, meta-learning resulted in a statistically significant improvement for 13 of 19 PKs ($p\text{-value} < 0.001$). By contrast, only one target (MELK) had a significant decrease in performance. The consistent improvements suggested that meta-learning had a further improved generalization capability compared to standard transfer learning. The absolute median increase in AUC as a consequence of meta-learning was generally small, ranging from 0.01 to 0.04 (Fig. 4A). However, this was partly a consequence of the broad AUC value distributions of independent trials. These distributions mirrored high variance when small PKI sets of different composition were used for fine-tuning. The consistency in improvements through meta-learning was much more evident when the difference in AUC (dAUC) was determined on a per-trial basis, as shown in Fig. 4B. For most of the PKs having a statistically significant difference, the boxes accounting for the interquartile range fell exclusively into the positive range, with only very few negative outliers.

Models for non-overlapping data sets of source and target compounds

Next, the models were re-generated after removal of shared source and target compounds (multi-PK inhibitors). Removal of shared compounds resulted in a significant reduction of the pre-training data $S^{(-t, -\cap)}$, ranging from 11,296 to 42,859 PK-PKI interactions, depending on the PK target. For non-overlapping compound sets, layer freezing strongly reduced the prediction accuracy of both deep learning models. Therefore, instead of retaining the learned patterns in a fixed bottleneck feature representation, fine-tuning was applied to all layers of the base model. Removal of shared compounds together with the ensuing reduction of training data further increased the difficulty of the prediction task. Therefore, we also included RF models in the analysis as a further reference. As anticipated, the performance of the re-generated models was reduced compared to the original models (Fig. 5). Median AUC values of the models ranged from ~ 0.65 to ~ 0.80 , depending on the method and PK target. Compared to standard transfer learning, meta-learning yielded statistically significant improvements in performance for 12 of 19 PKs (Fig. 5A). Standard transfer learning produced a statistically significant improvement for only 1 PK (HIPK2). Similarly, compared to RF, meta-learning resulted in statistically significant improvements in performance for 14 PKs (Fig. 5B). RF models produced significantly improved performance in only two cases (CDK8, BLK). Thus, models built for non-overlapping source and target compound sets also yielded a preferential gain in performance as a consequence of meta-learning. As observed before for models built in the presence of shared source and target compounds, statistically significant gains in performance were generally of small magnitude, but consistent. This was further revealed by dAUC comparison of meta- and standard transfer learning on a pair-trial basis (Fig. 6A) as well as meta-learning and RF models (Fig. 6B).

Negative transfer

We then investigated the potential of the meta-learning method to control negative transfer. Therefore, NTI values were computed according to Eq. 8. Here, $f(x; T, \zeta)$ represents the RF and $h(x; S, T, \psi)$ the standard transfer learning model. In Fig. 7, target PKs were arranged in the order of ascending NTI values comparing RF and standard transfer learning models. Mean AUC values are reported for different models based on non-overlapping sets of source and target compounds. Thus, the tendency of negative transfer increased from the left to the right.

For the first PKs on the left, standard transfer learning resulted in a large AUC increase compared to the RF model, which indicated the presence of positive transfer. In this case, the performance of the meta-learning and standard transfer learning models was very similar. With beginning negative transfer, the performance of the standard transfer learning model became increasingly similar to the RF model while meta-learning yielded substantial performance increases. For PKs with largest negative transfer, the RF model met and then exceeded the performance of the meta-learning and standard transfer learning models. However, in these cases, meta-learning reduced negative transfer by $\sim 50\%$ compared to standard transfer learning. Thus, in the presence of increasing negative transfer, meta-learning achieved highest performance or at least balanced negative transfer compared to standard transfer learning. These findings indicated that the meta-learning algorithm effectively increased the relative weight of source samples that were relevant for the target tasks and reduced the weight of others that caused negative transfer.

Conclusion

Machine learning in data-sparse domains generally benefits from the application of specialized learning concepts such as transfer or meta-learning, which rely on knowledge exchange between different domains. A pre-requisite for their application is the presence of related prediction tasks with varying amounts of available data to facilitate knowledge transfer to low-data tasks. Standard transfer learning typically relies on model pre-training in a source domain where sufficient data are available, followed by fine-tuning in the target domain with limited amounts of available data. On the other hand, meta-learning primarily aims at adapting models to new tasks without major training requirements, for instance, by learning of weight initializations. To this end, a distinguishing feature of meta-learning is the optimization of a meta-objective to ensure that models are readily adaptable.

We have reasoned that transfer and meta-learning are complementary in nature and might be integrated to further improve learning and prediction accuracy. Therefore, we have developed a new meta-learning algorithm that combines target protein (task) and compound (instance) information and acts as a front end of model pre-training for transfer learning. Accordingly, a model is pre-trained in a source domain using weights for compound-target interactions determined by the meta-learning model. This provides an advanced basis of fine-

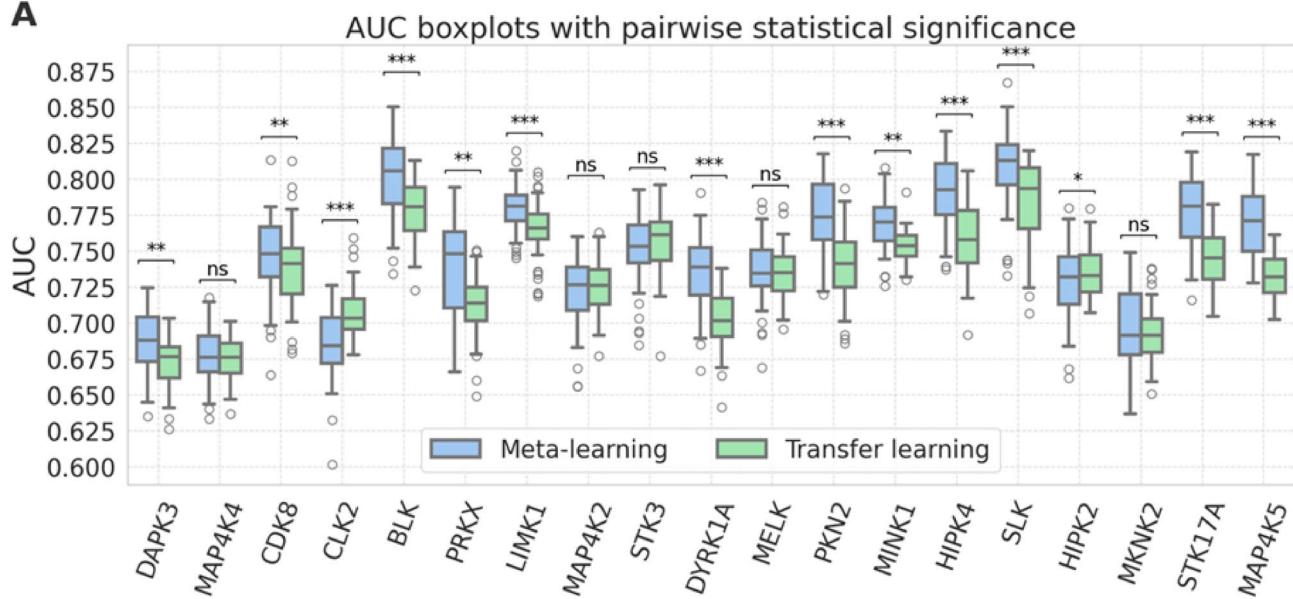
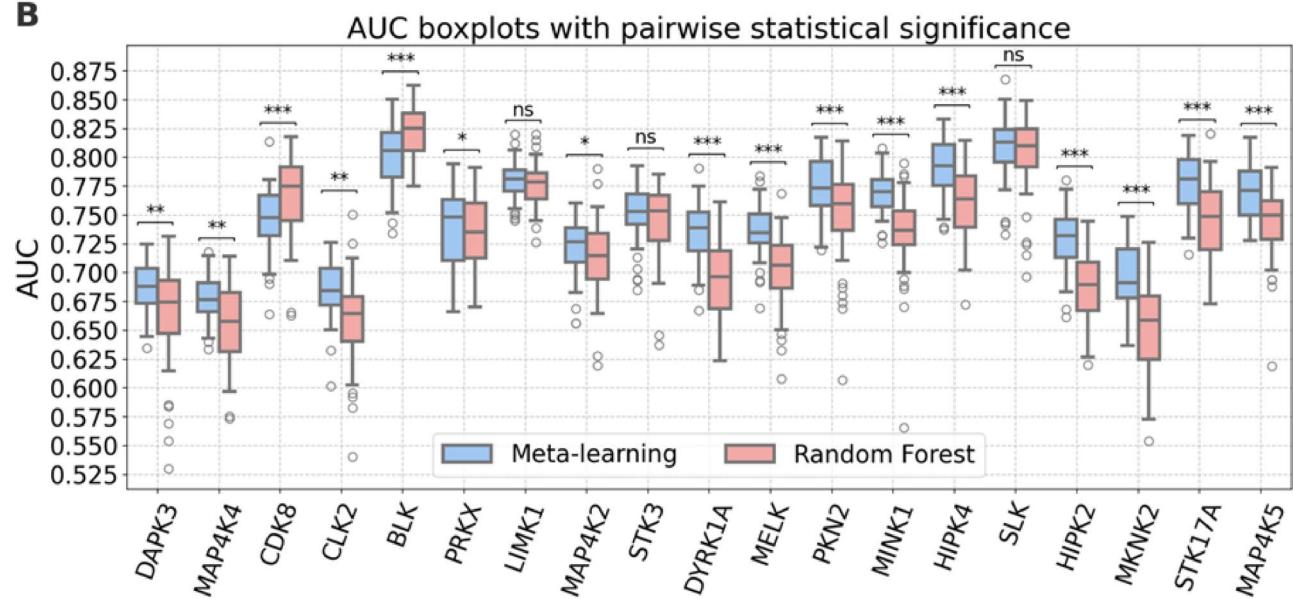
A**B**

Fig. 5. Model performance for the second calculation setting. (A) and (B) show AUC boxplots for 50 independent trials of meta-learning compared to standard transfer learning and RF models, respectively. Statistical significance of observed differences is reported according to Fig. 4.

tuning in a target domain. The meta-objective of the model is optimization of the generalization potential of the pre-trained model in the target domain. Therefore, the meta-learning algorithm identifies the most suitable subset of compounds for pre-training.

In our proof-of-concept application, meta-learning led to statistically significant improvements in prediction accuracy for the majority of tasks. For low-data applications, these improvements are relevant, given their consistency. Moreover, since the meta-learning algorithm also works at the instance level (leading to the identification of optimized training compound subsets), it is applicable to explicitly address the problem of negative transfer, a key limitation of standard transfer learning. We demonstrated that the use of the meta-model as a transfer learning front end effectively controlled negative transfer effects during the predictions. Therefore, the consistent improvements in prediction accuracy should also be considered in the context of balanced negative transfer, rendering the new approach more robust than standard transfer learning for any applications affected by negative transfer. Taken together, our findings indicate that the meta-learning algorithm introduced herein provides new opportunities for machine learning in low-data regimes. Although the methodology was primarily conceived for cheminformatics, it is generally applicable and can be adopted in other fields.

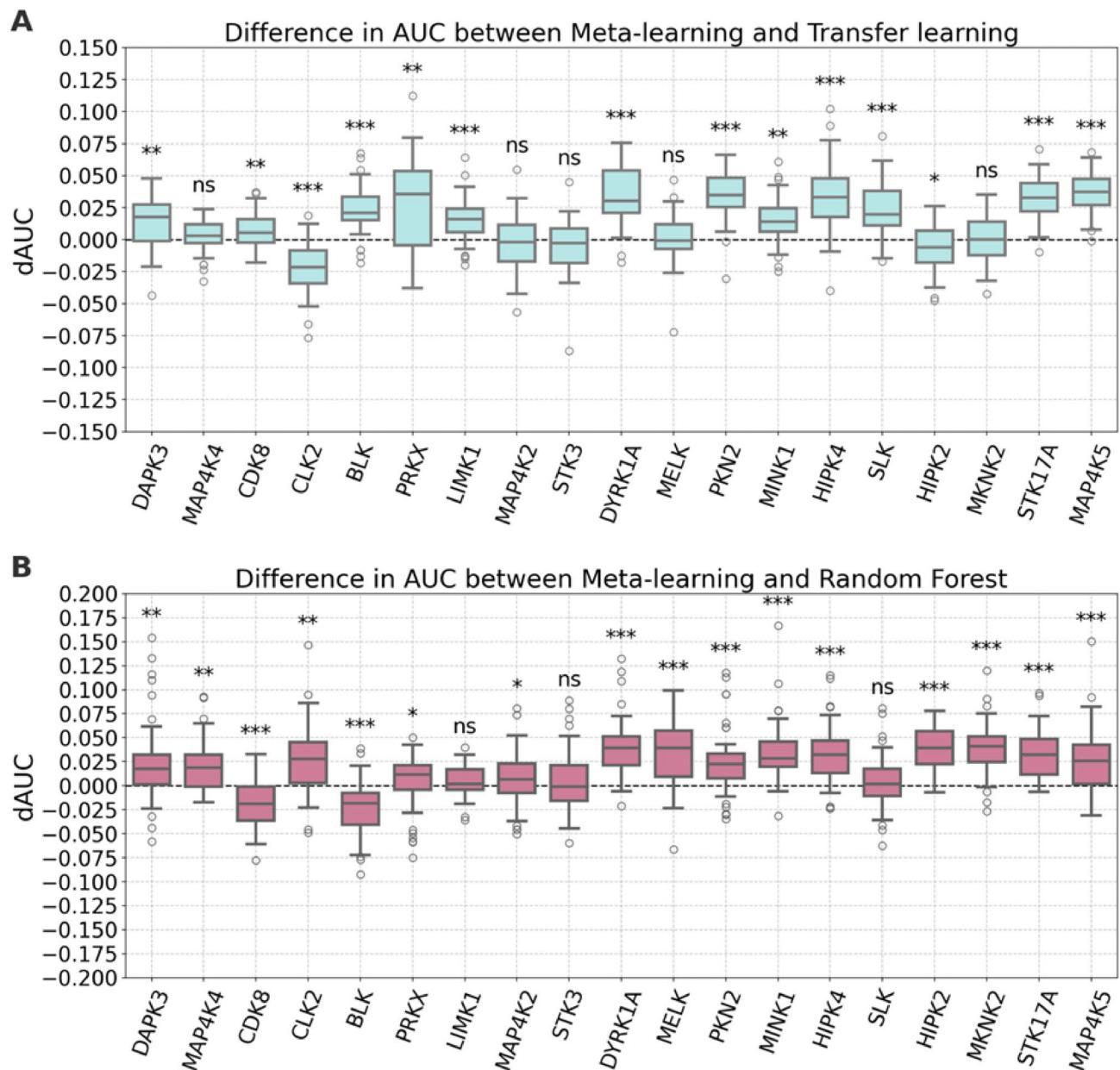


Fig. 6. Performance differences on a per-trial basis for the second calculation setting. Boxplots show the distributions of dAUC values determined for each individual trial of meta-learning compared to (A) standard transfer learning and (B) RF models. Positive values indicate improved performance of meta-learning compared to other models. Statistical significance is reported according to Fig. 4.

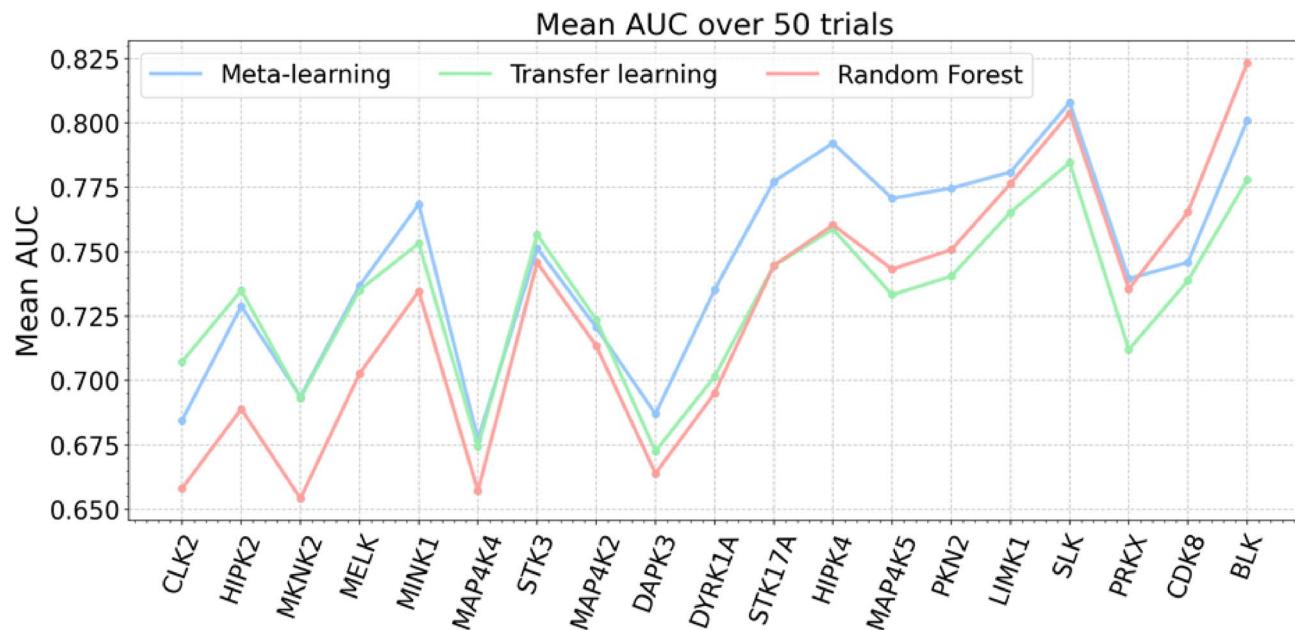


Fig. 7. Assessment of negative transfer. For models based on non-overlapping source and target compound sets, mean AUC values over 50 independent trials are reported. The 19 PK targets are sorted in the order of ascending NTI values comparing RF and standard transfer learning models.

Data availability

Data and code generated for this study are available via the following link: <https://uni-bonn.sciebo.de/s/LRP43AmLaWA5Xc6>.

Received: 1 July 2025; Accepted: 25 September 2025

Published online: 09 October 2025

References

1. Cai, C. et al. Transfer learning for drug discovery. *J. Med. Chem.* **63**, 8683–8694 (2020).
2. Ye, Z., Yang, Y., Li, X., Cao, D. & Ouyang, D. An integrated transfer learning and multitask learning approach for Pharmacokinetic parameter prediction. *Mol. Pharm.* **16**, 533–541 (2019).
3. Boonyarit, B. et al. GraphEGFR: Multi-task and transfer learning based on molecular graph attention mechanism and fingerprints improving inhibitor bioactivity prediction for EGFR family proteins on data scarcity. *J. Comput. Chem.* **45**, 2001–2023 (2024).
4. Li, H. et al. Improving molecular property prediction through a task similarity enhanced transfer learning strategy. *iScience* **25**, 105231 (2022).
5. Fooladi, H., Hirte, S. & Kirchmair, J. Quantifying the hardness of bioactivity prediction tasks for transfer learning. *J. Chem. Inf. Model.* **64**, 4031–4046 (2024).
6. Hospedales, T., Antoniou, A., Micaelli, P. & Storkey, A. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**, 5149–5169 (2021).
7. Kötter, A. et al. Task-similarity is a crucial factor for few-shot meta-learning of structure-activity relationships. *ChemBioChem* **25**, e202400095 (2024).
8. Shu, J. et al. Meta-Weight-Net: Learning an explicit mapping for sample weighting. Preprint at (2019). <https://arxiv.org/abs/1902.07379>
9. Freund, Y. & Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comp. Sys Sci.* **55**, 119–139 (1997).
10. Finn, C., Abbeel, P. & Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. Preprint at (2017). <https://arxiv.org/abs/1703.03400>
11. Chen, H. & Bajorath, J. Meta-learning for transformer-based prediction of potent compounds. *Sci. Rep.* **13**, 16145 (2023).
12. Deleu, T. & Bengio, Y. The effects of negative adaptation in model-agnostic meta-learning. Preprint at (2018). <https://arxiv.org/abs/1812.02159>
13. Jerfel, G., Grant, E., Griffiths, T. & Heller, K. A. Reconciling meta-learning and continual learning with online mixtures of tasks. *Adv. Neural Inf. Process. Syst.* **32**, 9122–9133 (2019).
14. Zhang, W., Deng, L., Zhang, L. & Wu, D. A survey on negative transfer. *IEEE/CAA J Autom. Sinica.* **10**, 305–329 (2022).
15. Wang, Z., Dai, Z., Póczos, B. & Carbonell, J. Characterizing and avoiding negative transfer. Preprint at (2019). <https://arxiv.org/abs/1811.09751>
16. Huan, Z. et al. Learning to select instance: simultaneous transfer learning and clustering. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* pp. 1950–1954. (2021).
17. Manning, G., Whyte, D. B., Martinez, R., Hunter, T. & Sudarsanam, S. The protein kinase complement of the human genome. *Science* **298**, 1912–1934 (2002).
18. Gaulton, A. et al. The ChEMBL database in 2017. *Nucleic Acids Res.* **45**, 945–954 (2017).
19. Liu, T., Lin, Y., Wen, X., Jorissen, R. N. & Gilson, M. K. BindingDB: A web-accessible database of experimentally determined protein-ligand binding affinities. *Nucleic Acids Res.* **35**, D198–D201 (2007).
20. Voßen, S., Xerxa, E. & Bajorath, J. Assessing darkness of the human Kinome from a medicinal chemistry perspective. *J. Med. Chem.* **67**, 17919–17928 (2024).

21. Weininger, D. SMILES, a chemical Language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **28**, 31–36 (1988).
22. RDKit. Cheminformatics and machine learning software (2013). <http://www.rdkit.org> (accessed on 2025-05-10).
23. Rogers, D. & Hahn, M. Extended-connectivity fingerprints. *J. Chem. Inf. Model.* **50**, 742–754 (2010).
24. Ziyin, L., Liu, K., Mori, T. & Ueda, M. Strength of minibatch noise in SGD. Preprint at (2022). <https://arxiv.org/abs/2102.05375>
25. Srivastava, N. et al. A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
26. Gal, Y. & Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. Preprint at (2016). <https://arxiv.org/abs/1506.02142>
27. Seoh, R. Qualitative analysis of Monte Carlo dropout. Preprint at (2020). <https://arxiv.org/abs/2007.01720>
28. Verdoja, F. & Kyrki, V. Notes on the behavior of MC dropout. Preprint at (2021). <https://arxiv.org/abs/2008.02627>
29. Pedregosa, F. et al. Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
30. Ba, J. L., Kiros, J. R. & Hinton, G. E. Layer Normalization. Preprint at (2016). <https://arxiv.org/abs/1607.06450>
31. Soekhoe, D., van der Putten, P. & Plaat, A. On the impact of data set size in transfer learning using deep neural networks. In *Advances in intelligent Data Analysis* (eds Boström, X. V. et al.) 50–60 (Springer International Publishing, 2016).
32. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. Preprint at (2014). <https://arxiv.org/abs/1412.6980>
33. Paszke, A. et al. An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **32**, 8026–8037 (2019).
34. Brown, C. D. & Davis, H. T. Receiver operating characteristics curves and related decision measures: A tutorial. *Chemometrics Intell. Lab. Syst.* **80**, 24–38 (2006).
35. Hu, Y., Furtmann, N. & Bajorath, J. Current compound coverage of the Kinome. *J. Med. Chem.* **58**, 30–40 (2015).
36. Knight, J. D. R., Qian, B., Baker, D. & Kothary, R. Conservation, variability and the modeling of active protein kinases. *PLOS One.* **2**, e982 (2007).
37. Du, M., Xie, X., Luo, J. & Li, J. Meta-learning-based inductive logistic matrix completion for prediction of kinase inhibitors. *J. Cheminf.* **16**, 44 (2024).
38. Ren, Q. et al. KinomeMETA: Meta-learning enhanced kinome-wide polypharmacology profiling. *Brief. Bioinform.* **25**, bbad461 (2023).

Author contributions

All authors contributed to designing and conducting the study, analyzing the results, and preparing the manuscript.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1038/s41598-025-22058-3>.

Correspondence and requests for materials should be addressed to J.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2025